TRAFFIC DATA PLATFORM BASED ON THE SERVICE ORIENTED ARCHITECTURE (SOA)

TOUKO TCHEUMADJEU Louis Calvin, German Aerospace Center (DLR) - Institute of Transportation Systems, Rutherfordstr.2, 12489 Berlin – Germany, Louis.ToukoTcheumadjeu@DLR.de

RUPPE Sten, German Aerospace Center (DLR) - Institute of Transportation Systems, Rutherfordstr.2, 12489 Berlin – Germany, Sten.Ruppe@DLR.de

BROCKFELD Elmar, German Aerospace Center (DLR) - Institute of Transportation Systems, Rutherfordstr.2, 12489 Berlin – Germany, Elmar.Brockfeld@DLR.de

YAHYAOUI Younes, German Aerospace Center (DLR) - Institute of Transportation Systems, Rutherfordstr.2, 12489 Berlin – Germany, Younes.Yahyaoui@DLR.de

ABSTRACT

The DLR Traffic Data Platform (TDP) that is currently being developed by the German Aerospace Center (DLR) is an autonomous decentralised ITS system for distributed intelligent traffic data management and dissemination. Of course, there are many possibilities to design the architecture of such a traffic data platform where service oriented architecture (SOA) has been chosen for the current design of the TDP. In this paper the SOA design aspect of the TDP will be analysed and presented. The TDP as service provider is able to store, manage, aggregate, process and fuse traffic data from different sources like floating car data (FCD), loop detector data, remote sensing, video sensors and traffic relevant radar data into common traffic states and supply them as services to the TDP clients or service consumers who need these data to realize telematic services. The TDP is designed to support not only “online” traffic information but also services like multimodal routing for example. Having all these data in a single platform and making it available by providing standardised ubiquitous access, research and the development of new methods as well as the enhancement of existing methods such as data fusion, traffic state estimation, quality evaluations and innovative telematic services could be facilitated. Thus, the TDP contributes to more effective research in the field of traffic management.

Keywords: traffic data collection, floating car data, probe vehicle data, loop detector, data fusion, travel time, traffic model, congestion management; video, remote sensing, service oriented architecture

INTRODUCTION AND MOTIVATION

The important role that traffic data platforms play in the ITS world today should not be neglected. Traffic data platforms are ITS infrastructure for intelligent traffic data management and dissemination that constitute an advantage for the realisation of ITS applications and telematic services as well as for research purposes. There are many possibilities to design the architecture of such a traffic data platform that generally consists of several autonomous and heterogeneous modules. Usually traffic data which are stored and processed are from loop detectors, video sensors, TMC services and, quite recently, mobile sources like floating car data (FCD) [1] or floating phone data (FPD).
The requirements concerning research with such a traffic data platform are even larger. Not only currently generated and integrated traffic state information are of interest, but also the handling of raw data and data from new upcoming data sources like remote sensing, new video sensors [2] and radar need to be supported. Furthermore, access to historical data should be possible. The standardised access and availability of these data, as well as the reusability of existing methods allow for more effective research in the field of traffic management and facilitate the development of new methods as well as the enhancement of existing methods.
For this purpose the German Aerospace Center (DLR) Institute of Transportation Systems is developing a modular traffic data platform (**DLR Traffic Data Platform**) that is able to process and fuse traffic data from different sources mentioned above into common traffic states (cf. [3]). The TDP is currently built as a service-oriented architecture (SOA) which allows a simple integration of new services based on provided traffic data into the system. Using an open and flexible API for integration of services, new data sources and applications can be added easily to the TDP.

This contribution gives a general overview about the internal structured layering of the TDP. Next, it describes the model architecture of the TDP and introduces shortly the SOA technology. Finally, the SOA design architecture for the TDP will be analysed and presented.

THE TDP MODULAR ARCHITECTURE

Figure 1 shows an overview about the internal structure of the modular architecture of the Traffic Data Platform as designed by the DLR. Traffic data (raw data) as well as road network information (digital map) are imported into the platform using an open and flexible interface. Internally, many processes like data parsing, filtering, processing, aggregation, quality evaluation, data fusion and storage into the database are required for the intelligent management of the imported data. The resulting traffic information such as traffic flow, travel time and traffic states as well as the original data can be retrieved by client applications for large scale traffic demand estimation or short-time adaptive traffic control for example. Moreover, the TDP is designed to support not only "online" information but also telematic services like dynamic multimodal routing as shown in Figure 1. For the system configuration, administration and monitoring the TDP management console has been developed as web based graphical user interface. The TDP contains also authentication and authorization features for the management of client access and resources and provides flexible APIs for easy integration of services, new data sources and applications to the platform.

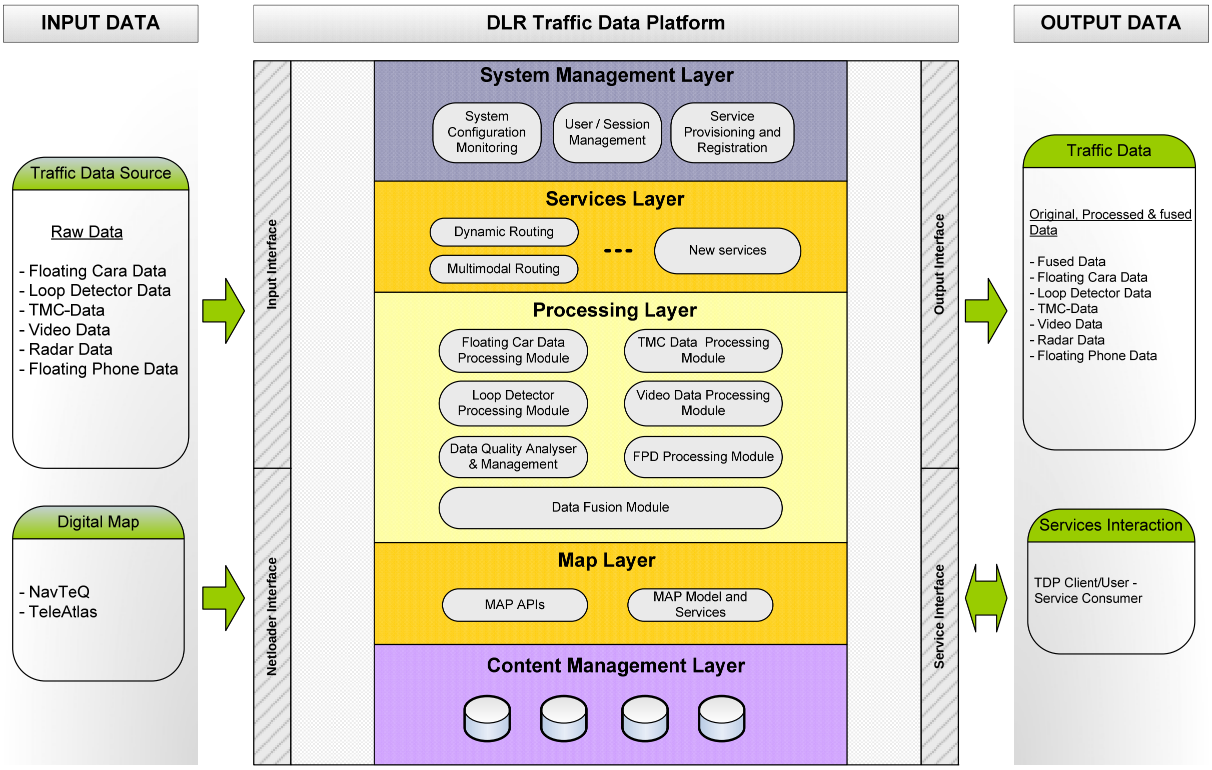

Figure 1: The modular architecture of TDP by layers

THE TDP MODEL ARCHITECTURE

In this section the model architecture of two TDP layers are described exemplarily - the processing layer and the content management layer. As shown in Figure 2, the modules of the processing layer consist of processing units like FCD processing chain units, loop processing chain units, data fusion units etc. Each processing unit is an autonomous decentralised component that processes and enhances input data as well as data selected from the available data pool. A processing unit can use the results from other processing and service units.

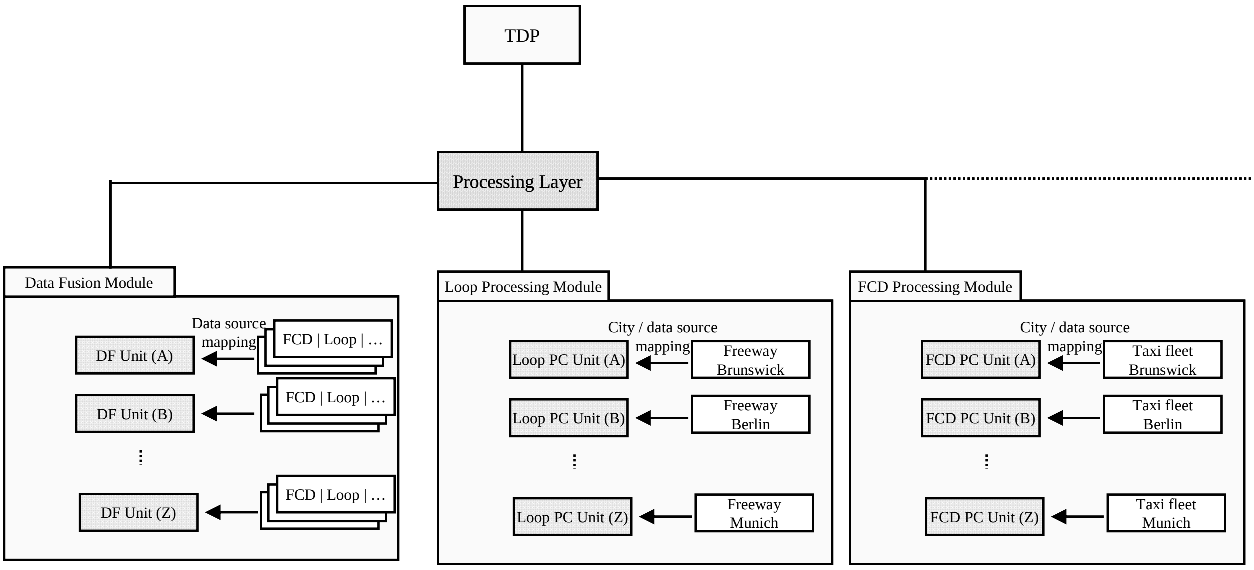

Figure 2: TDP Processing Layer

As shown in Figure 3, the content management layer consists of a database connector module and a FTP connector module for very large data that provide resources for data storage, extraction and management. TDP is designed to support different relational database management systems like MySQL, PostgreSQL and Oracle.

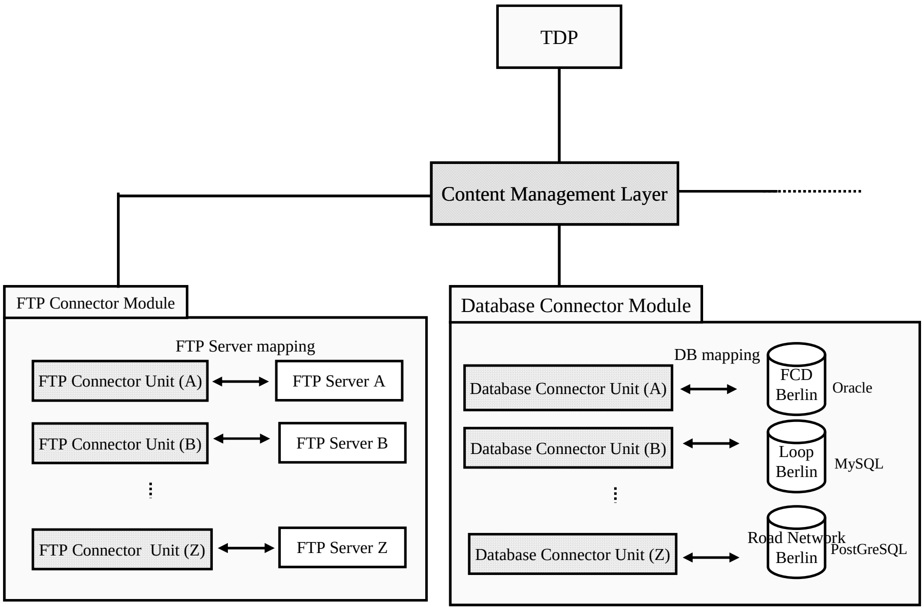

Figure 3: TDP Content Management Layer

One typical TDP workflow consists of three steps:

- Data import,
- Data processing, enhancement and storage,
- Data export.

The FCD processing chain unit, considered as basic component of the TDP and also as one of many data processing components within the TDP, has been chosen to demonstrate the design and service oriented architecture approach in this paper. Because of the modular decentralised architecture of the TDP, the SOA design of the FCD processing chain unit can be easily translated to other processing components of TDP like loop detector or video data processing modules.

The FCD processing chain unit is responsible for filtering, aggregation and processing of raw floating car data to provide link based travel speeds as well as travel times. The FCD processing chain unit consists of four main components as illustrated in Figure 4:

1. *Filter:* component of the FCD processing module, implements some basic rules to detect and eliminate incorrect input data from further processing. Such data are still stored as raw data into the corresponding database, but are marked by a flag.

2. *Trajectoriser:* detects vehicle trajectories from single GPS points using the vehicle IDs of a given FCD system.

3. *Matcher:* computes map-matched positions from trajectories of GPS points based on enhanced algorithms [4, 5]. In this context, current traffic state information provided by the link speed generator component is dynamically used to obtain optimal matching results.

4. *LinkSpeedGenerator:* provides relevant traffic state information for all links of a considered road network based on current (and historical) floating car data in real-time. In doing so, it enables the matcher component to generate optimal results.

The FCD Processing Chain Unit has also a model for storage of all configuration parameters and an input connector as interface to receive raw data forwarded by the corresponding data importer. Database and FTP connectors are used to store the processed data into the corresponding database or into a file system.

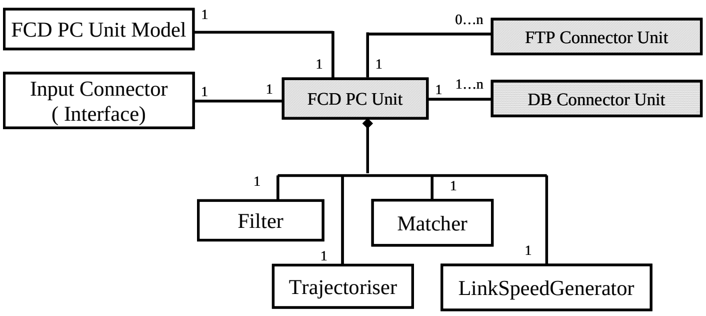

Figure 4: FCD processing chain unit model

TDP AS TRAFFIC DATA POOL

One of the main functionalities of TDP is the persistent storage and archiving of traffic data from different sources into distributed databases that have been previously configured for this purpose. In addition, the TDP provides resources to transform and load data into the database as well as to manage and to retrieve data. The TDP can be seen as a traffic data pool or data warehouse (see the Figure 5), and all traffic data managed by the TDP can theoretically be published as services to each stakeholder who needs traffic information or data for his own purposes. A big challenge for the TDP will be to manage and store the huge quantity of data from different types and to handle requests of many clients who are connected to the platform to retrieve online traffic data at the same time.

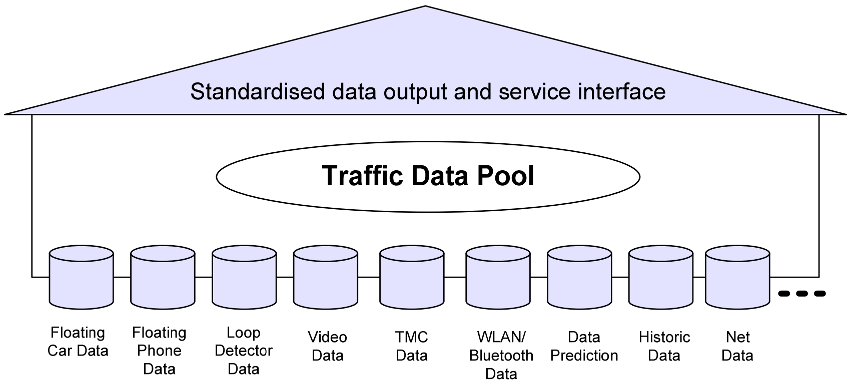

Figure 5: TDP as data warehouse

TDP AS RESEARCH PLATFORM

The Traffic Data Platform is designed not only for traffic data and services provision but can also serve as framework to reuse and improve existing methods as well as develop new methods, algorithms or to test new telematic services. Because of the possibility to have direct access to traffic data and services modules of different types, researchers will have opportunities of more effective research in the field of traffic management. Research activities can be placed within the platform (see the Figure 6 as an example of the integration of a data fusion module), as well as outside of the platform.

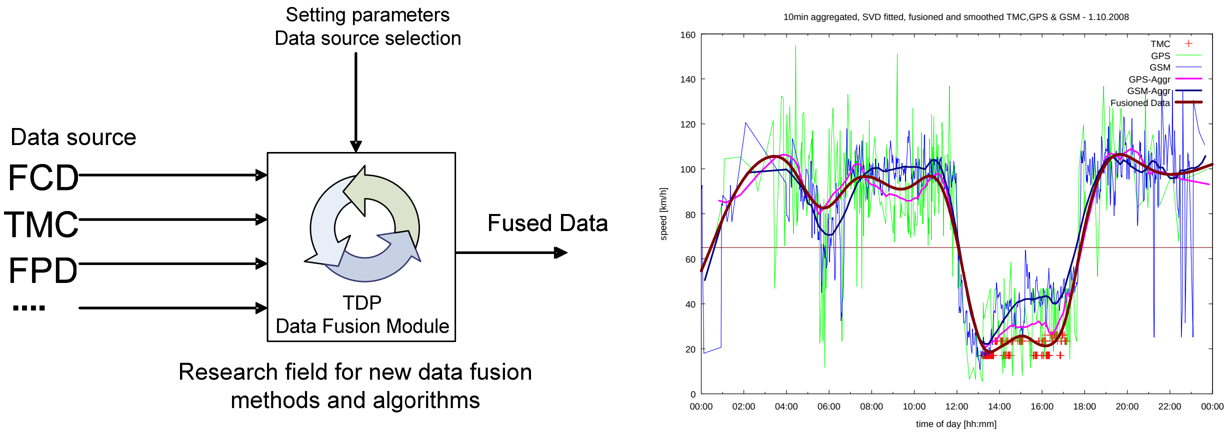

Figure 6: TDP as better place to conduct research for data fusion

SOA DESIGN ARCHITECTURE OF TDP

SOA is an architectural approach to create systems built from autonomous services (see Figure 7). To use the advantages of SOA (e.g. flexibility, robustness, service loose coupling, service abstraction and service reusability), the TDP is currently designed and built as service oriented architecture. As a consequence, all functionalities of the TDP as distributed data management and dissemination tool have been modelled as services. The role of TDP as ***service provider*** consists of making traffic data and telematic services available to TDP clients or ***service consumers***.

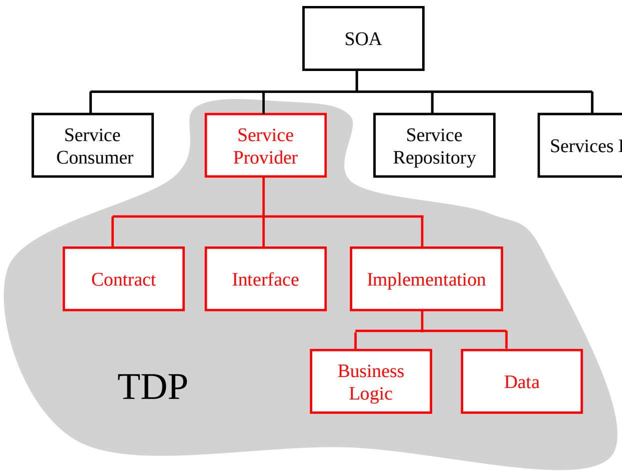

Figure 7: SOA components [6]

There are several ways to build SOA (e.g. CORBA, .NET, Web services …), TDP services is designed and implemented as web services. As shows in Figure 7 a service in SOA consists of following elements: contract, interface and implementation. The implementation element encapsulates business logic as well as the data. All these services will be published in a service registry (e.g. the Universal Description, Discovery Integration - UDDI). The service bus is a SOA infrastructure for the enterprise integration of defined services.
A service's contract is generally defined using Web Service Description Language (WSDL), while contracts for composition of services can be defined using Business Process Execution Language (BPEL) which, in turn, presents each web services as partner link. With service orientation, the business functionality encapsulated through the service can be used by any consumer outside of the control of the application or outside the company's firewall.

The basic SOA model as SOA triangle

Figure 8 shows the basic SOA model also called SOA triangle. The SOA triangle describes the interaction between the three main elements of SOA (Service Provider, Service Consumer or Requester and the Service Registry) as mentioned in Figure 7. In the basic SOA model the service provider implements a given service and publishes the service description in a service registry. The service consumer queries the registry to find a certain service. If found, it retrieves the location of the service and binds to the service endpoint, where the consumer can finally invoke the operations of the service. The communication with the web service is realised through the Simple Object Access Protocol (SOAP).

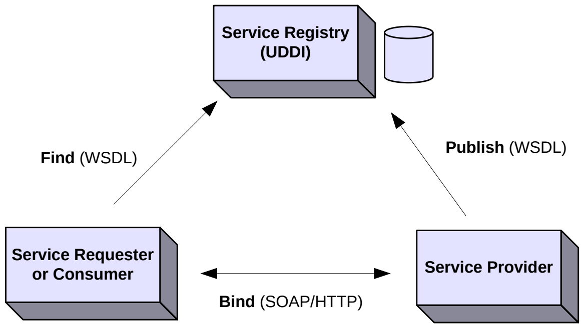

Figure 8: SOA triangle adapted to web services

In the SOA world different categories of services exist. The four types of services are summarised in the table below:

Table 1: Classification of services in SOA [6]

Nr	Service type	Description
1	Basic services	Basic services constitute the basic element of SOA. Basic services can be divided in data and logic centric services
2	Composed services	Composed services are stateless and can be divided in technology gateways, adapters, facades and functionality complementary services
3	Process oriented services	Encapsulate the business logic of the business process and are both client and server of a SOA and manage the process status
4	Public services	Public services provide interfaces for enterprise integration with extern application.

Corresponding to these four types of services, four services layers also exist in the SOA that map the different type of services mentioned above:

1. Enterprise Layer: consists of application front-end like a graphical user interface (GUI) or Batch-process. The element of this layer can instantiate all business processes and get the results.
2. Process Layer: contains process-oriented services
3. Composed Layer: contains the composed services
4. Basic layer: contains basic services

Based on the above classification of services, some examples of TDP functionalities modelled as services are described in the next section.

TDP Data Import – Input Interface

The input interface to import data from different sources in the TDP has been modelled as illustrated in the Figure 9. The TDP data importer as basic service defines different communication ways or protocols modelled as composed services like Web Services Gateway, FTP Gateway to import data from any TDP data provider.

In cases of system failure or malfunction of a subsystem like the FCD processing chain unit, TDP must be able to report cause and origin of the fault or perturbation in a logfile or the database. Moreover, an automatic e-mail message or short message service (SMS) has to be sent to the system administrator. In TDP, the Management Console Business Services like logging and tracing services for example fulfil these tasks.
The security aspect has been taken into account by the TDP. The TDP provides authentication and authorization services as security feature to authenticate and authorise each TDP client (Data provider and requester), that interacts with the TDP system.

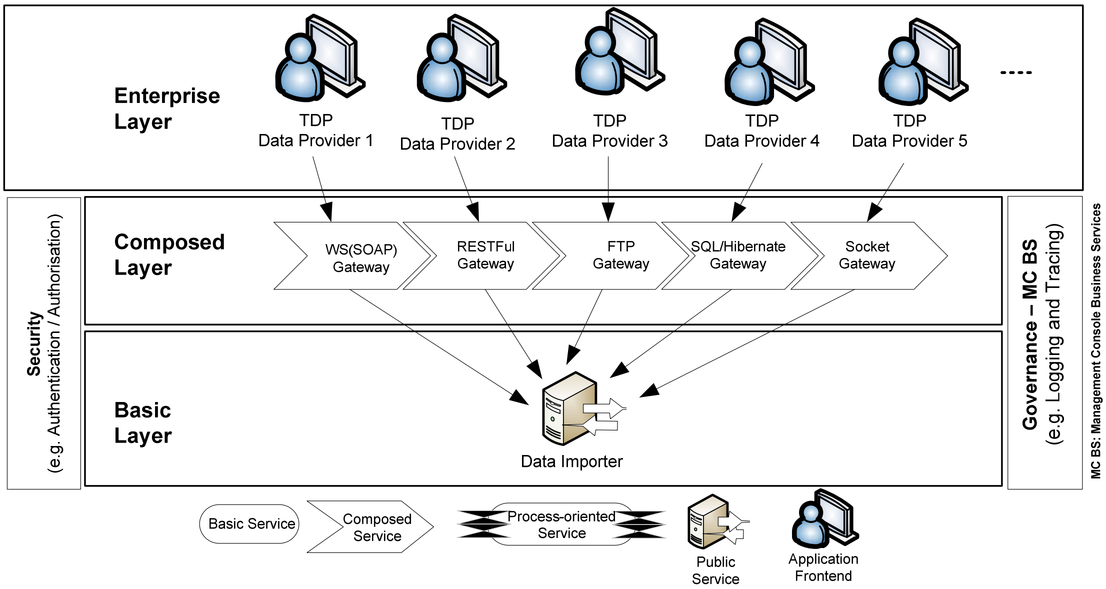

Figure 9: Services classification for TDP data import

TDP FCD Processing Chain Unit

In Figure 10, the FCD processing chain unit as process oriented service periodically receives raw FCD data from the data importer. For processing the received data four basic services are implemented (Filter, Trajectoriser, Matcher, LinkSpeedGenerator). Two composed services are introduced: the Facade *Trajectoriser-Matcher-LinkSpeedGenerator* which defines an API that hides the interaction complexity between the three services and the database connector that can be used as interface for data storage into and extraction from the database. The Map service is used by the matcher basic service during the map matching process.

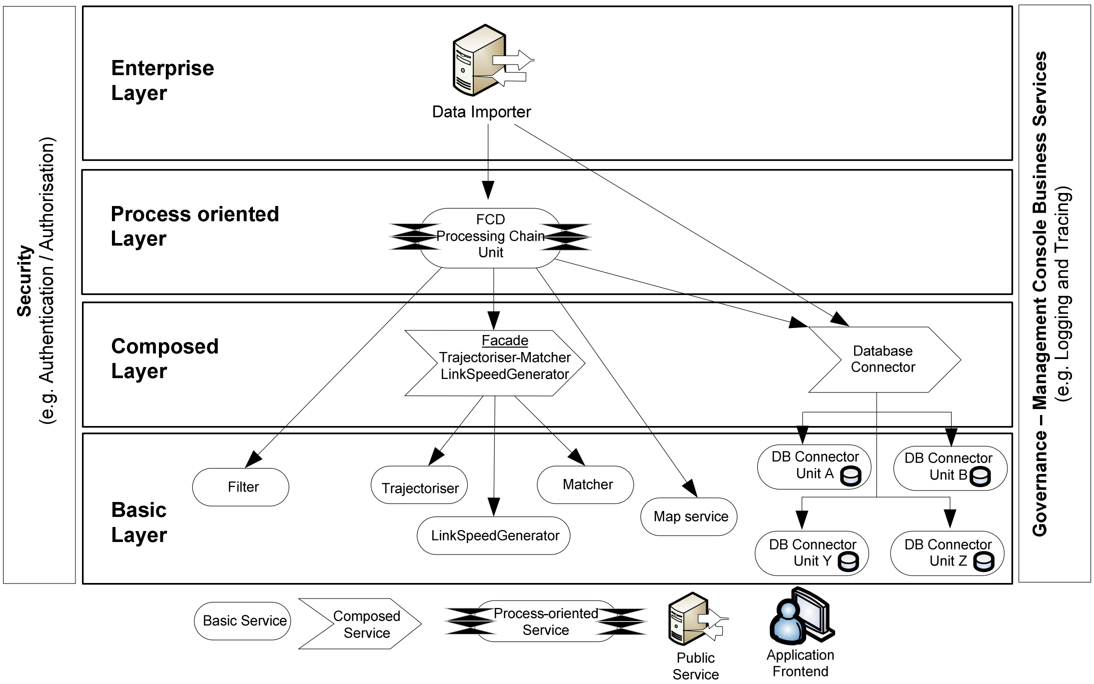

Figure 10: Services classification of FCD processing chain unit

Figure 11 depicts the business process modelling of the process oriented services called FCD processing chain unit. The internal data flow and processing task of Data importer, Filter, Trajectoriser, Matcher and LinkSpeedGenerator as services have been described in detail.

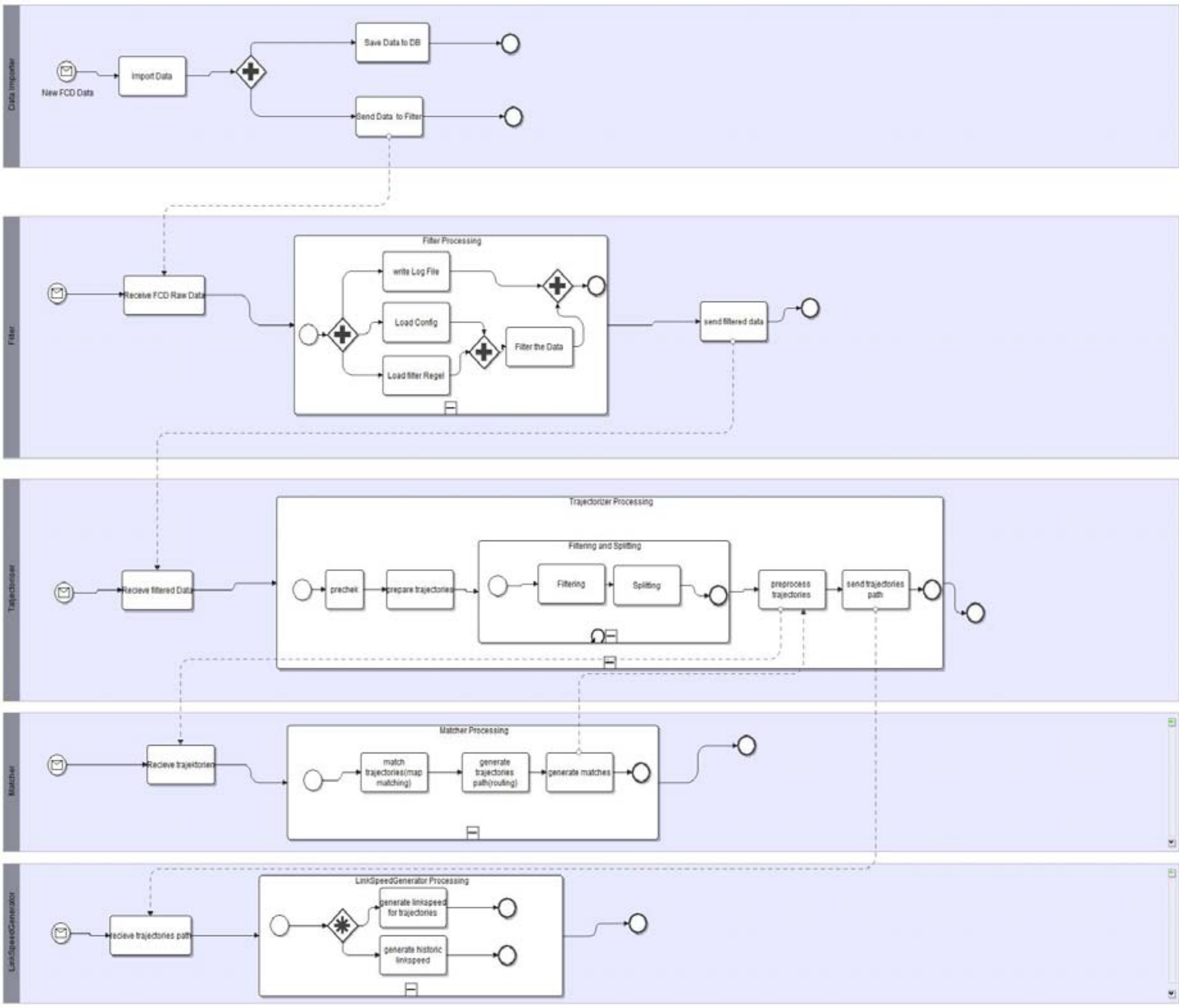

Figure 11: Business process modelling of FCD processing chain unit

TDP Data Export – Output Interface

The output interface to export data (original data, archive data as well as processed data) from TDP has been modelled as described in Figure 12. The TDP data exporter as basic service defines different communication ways or protocols as composed services like the Web Services Gateway and the FTP Gateway to export the data to the TDP clients or service consumers. The aim of TDP is to achieve a system solution for every client profile by providing ubiquitous client access to the TDP.

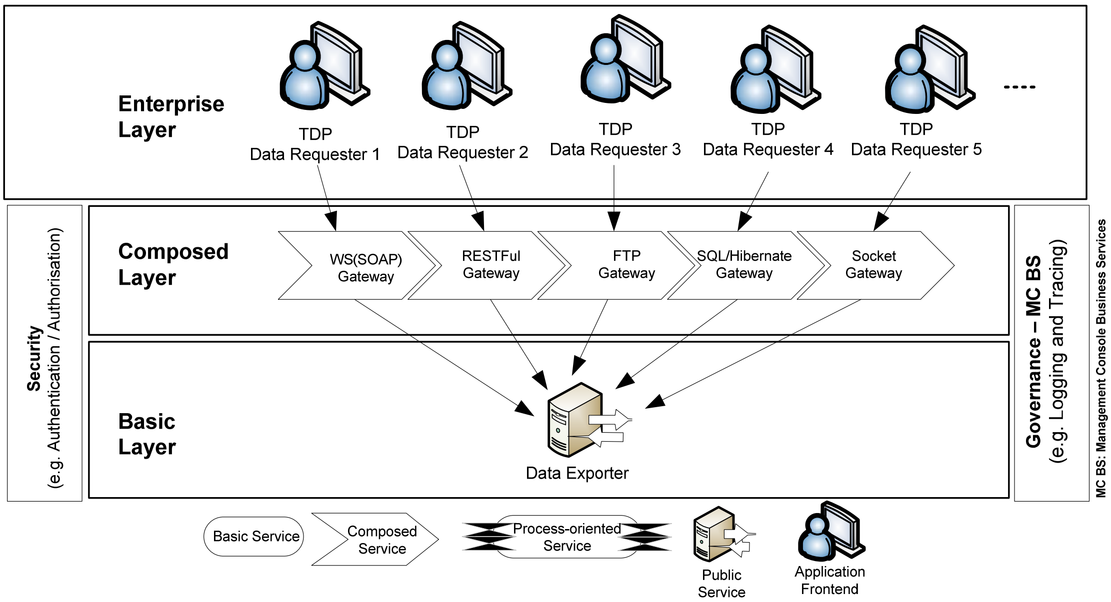

Figure 12: Services classification for data export

Implementation of TDP FCD processing chain unit using Enterprise Services Bus (ESB)

After the aforementioned identification and classification of TDP functionalities as services, the enterprise service bus (ESB) has been used to realise the integration of all defined TDP services with service requesters in a SOA environment. Figure 13 illustrates an implementation of an FCD processing chain unit as enterprise service bus. An enterprise service bus is a logical architectural component that provides an integration infrastructure for services-oriented architecture. The advantage of ESB is its ability to support incremental service and application integration. Some of the key features or capabilities [7] of ESB are communication (e.g. routing, addressing), services interaction (e.g. services directory and discovery), service integration (e.g. service mapping, protocol transformation), quality of services, security, service level, management and autonomic, message processing and modelling.

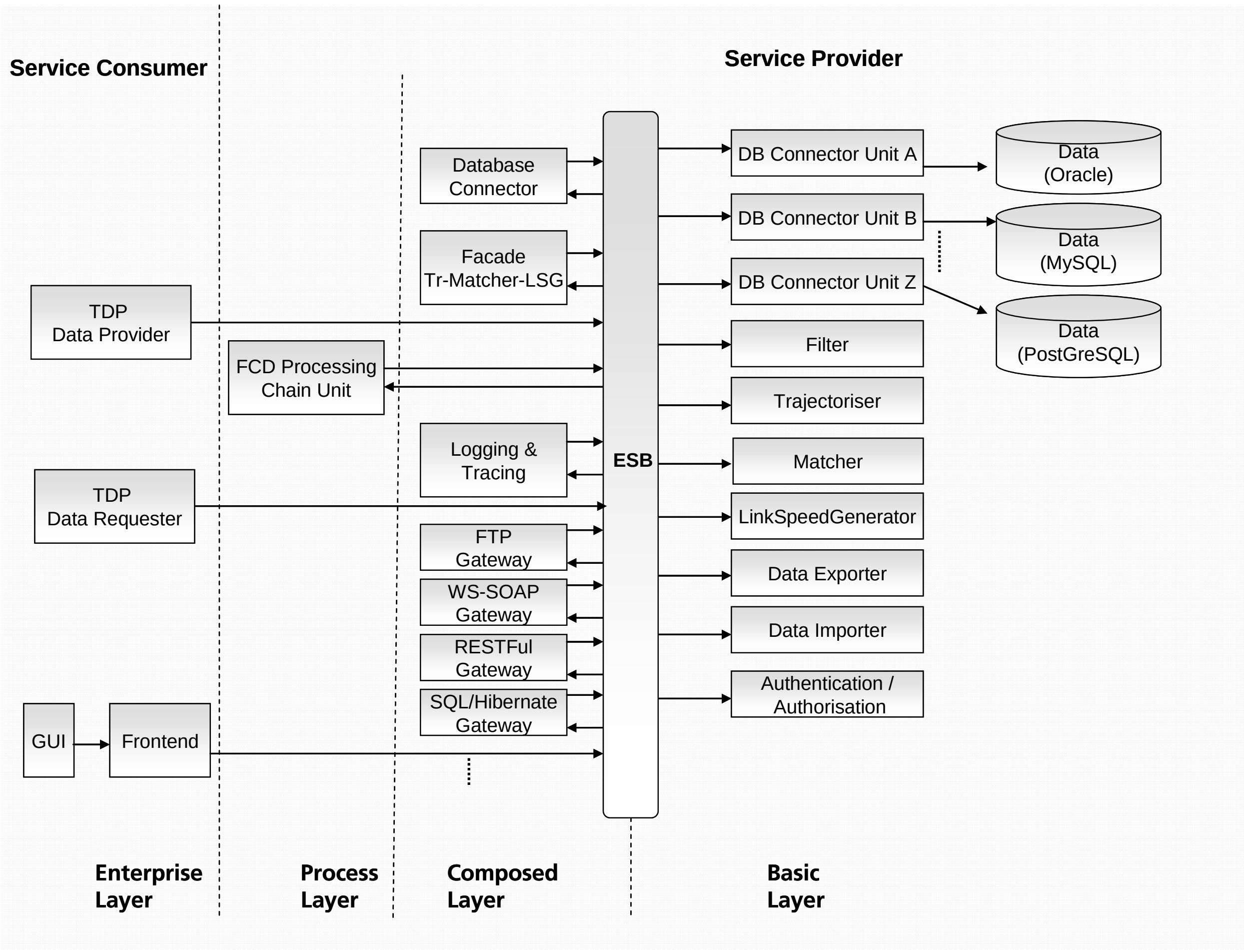

Figure 13: FCD processing chain unit implementation using ESB

CONCLUSION

The DLR Traffic Data Platform (TDP) presented in this paper is a complex modular and decentralised system that consists of several autonomous distributed components and modules for storing, managing, aggregating, processing and fusing traffic data from different sources. In this context, TDP has been designed and built as services oriented architecture where SOA is defined as an architectural approach for creating systems built from autonomous services. To benefit from this SOA approach, each module of TDP is modelled as a service that provides specific functionalities (services). The main focus of this contribution is the design of a TDP architecture that fulfils the requirements of the SOA methodology and its analysis with respect to applications in traffic management.

REFERENCES

[1] Ralf-Peter Schäfer, Kai-Uwe Thiessenhusen, Elmar Brockfeld, Peter Wagner, "A traffic information system by means of real-time floating-car data", ITS World Congress 2002, Chicago (USA)

[2] Thomas, Ulrike und Rosenbaum, Dominik und Kurz, Franz und Suri, Sahil und Reinartz, Peter. „ A new Software/Hardware Architecture for Real Time Image Processing of Wide Area Airborne Camera Images ".Journal of Real-Time Image Processing , Band 4 (3) , Pages 229-244, Springer. August 2009

[3] Touko Tcheumadjeu,Louis Calvin; Ruppe, Sten; Brockfeld, Elmar; Wagner Peter. "Unified and Modular Traffic Data Platform". 16th ITS World Congress, September 21-25, 2009. Stockholm, Sweden

[4] Ebendt, Rüdiger; Sohr, Alexander; Touko Tcheumadjeu, Louis Calvin; Wagner, Peter. "Utilizing historical and current travel times based on floating car data for management of an express truck fleet", 5th International Scientific Conference: Theoretical and Practical Issues in Transport, pages 283-291, Feb. 11-12, 2010, Pardubice, Czech Republic

[5] Ebendt, Rüdiger; Wagner, Peter. "An Integrative Approach to Light- and Heavy-Weighted Route Planning Problems", 5th IMA Conference on Mathematics in Transport, Apr. 12-14, 2010, London, United Kingdom

[6] Dirk kraftzig; Karl Banke and Dirk Slama. "Enterprise SOA – Best Practices". Book, page 76, Mitp (2007) ISBN: 978-3-8266-1729-4

[7] Martin Keen and all. "Patterns: implementing an SOA Using an Enterprise Service Bus". IBM Redbooks. First Edition, July 2004. Pages 82-83